Straight lines have to be straight: automatic calibration and removal of distortion from scenes of structured enviroments

Frédéric Devernay, Olivier Faugeras

Machine Vision and Applications manuscript No.
(will be inserted by the editor)

Straight Lines Have to Be Straight

Automatic Calibration and Removal of Distortion from Scenes of Structured Environments

Frédéric Devernay, Olivier Faugeras

INRIA, BP93, 06902 Sophia Antipolis Cedex, e-mail: devernay,faugeras@sophia.inria.fr

The date of receipt and acceptance will be inserted by the editor

Most algorithms in 3-D Computer Vision rely on the pinhole camera model because of its simplicity, whereas video optics, especially low-cost wide-angle or fish-eye lens, generate a lot of non-linear distortion which can be critical.

To find the distortion parameters of a camera, we use the following fundamental property: a camera follows the pinhole model if and only if the projection of every line in space onto the camera is a line. Consequently, if we find the transformation on the video image so that every line in space is viewed in the transformed image as a line, then we know how to remove the distortion from the image.

The algorithm consists of first doing edge extraction on a possibly distorted video sequence, then doing polygonal approximation with a large tolerance on these edges to extract possible lines from the sequence, and then finding the parameters of our distortion model that best transform these edges to segments.

Results are presented on real video images, compared with distortion calibration obtained by a full camera calibration method which uses a calibration grid.

1 Introduction

1.1 External, internal, and distortion calibration

In the context of 3-D computer vision, camera calibration consists of finding the mapping between the 3-D space and the camera plane. This mapping can be separated in two different transformation: first, the displacement between the origin of 3-D space and the camera coordinate system, which forms the external calibration parameters (3-D rotation and translation), and second the mapping between 3-D points in space and 2-D points on the camera plane in the camera coordinate system, which forms the internal camera calibration parameters.

The internal camera calibration parameters depend on the camera. In the case of an orthographic or affine camera model, optic rays are all parallel and there are only 3 parameters corresponding to the spatial sampling of the image plane. The perspective (or projective) camera model involves two more camera parameters corresponding to the position of the principal point in the image (which is the intersection of the optical axis with the image plane). For many applications which require high accuracy, or in cases where low-cost or wide-angle lenses are used, the perspective model is not sufficient and more internal calibration parameters must be added to take into account camera lens distortion.

The distortion parameters are most often coupled with internal camera parameters, but we can also use a camera model in which they are decoupled. Decoupling the distortion parameters from others can be equivalent to adding more degrees of freedom to the camera model.

1.2 Brief summary of existing related work

Here is an overview of the different kinds of calibration methods available. The goal of this section is not to do an extensive review, and the reader can find more information in [3,18,22].

The first kind of calibration method is the one that uses a calibration grid with feature points whose world 3-D coordinates are known. These feature points, often called control points, can be corners, dots, or any features that can be easily extracted for computer images. Once the control points are identified in the image, the calibration method finds the best camera external (rotation and translation) and internal (image aspect ratio, focal length, and possibly others) parameters that correspond to the position of these points in the image. The simplest form of camera internal parameters is the standard pinhole camera [13], but in many cases the distortion due to wide-angle or low-quality lens has to be taken into account [26,3]. When the lens has a non-negligible distortion, using a calibration method with a pinhole camera model may result in high calibration errors.

The problem with these methods that compute the external and internal parameters at the same time arises from the fact that there is some kind of coupling between internal and external parameters that result in high errors on the camera internal parameters [27].

Another family of methods is those that use geometric invariants of the image features rather than their world coordinates, like parallel lines [6,2] or the image of a sphere [19].

The last kind of calibration techniques is those that do not need any kind of known calibration points. These are also called self-calibration methods, and the problem with these methods is that if all the parameters of the camera are unknown, they are still very unstable [12]. Known camera motion helps in getting more stable and accurate results [24,15] but it's not always that easy to get "pure camera rotation".

A few other calibration methods deal only with distortion calibration, like the plumb line method [5]. Another method presented in [4] uses a calibration grid to find a generic distortion function, represented as a 2-D vector field.

1.3 Overview of our method

Since many self-calibration [12] or weak calibration [29] techniques rely on a pinhole (i.e. perspective) camera model, our main idea was to calibrate only the image distortion, so that any camera could be considered as a pinhole camera after the application of the inverse of the distortion function to image features. We also don't want to rely on a particular camera motion [24] in order to be able to work on any kind of video recordings or snapshots (e.g. surveillance video recordings) for which there can be only little knowledge on self-motion, or some observed objects may be moving.

The only constraint is that the world seen though the camera must contain 3-D lines and segments. It can be city scenes, interior scenes, or aerial views containing buildings and man-made structures. Edge extraction and polygonal approximation is performed on these images in order to detect possible 3-D edges present in the scene, then we look for the distortion parameters that minimize the curvature of the 3-D segments projected to the image.

After we find a first estimate of the distortion parameters, we perform another polygonal approximation on the corrected (un-distorted) edges, this way straight line segments that were broken into several line segments because of distortion become one single line segment, and outliers (curves that were detected as line segments because of their small curvature) are implicitly eliminated. We continue this iterative process until we fall into a stable minimum of the distortion error after the polygonal approximation step.

In section 2, we review the different nonlinear distortion models available, including polynomial and fish-eye models, and the whole calibration process is fully described section 3.

2 The nonlinear distortion model

The mapping between 3-D points and 2-D image points can be decomposed into a perspective projection and a function that models the deviations from the ideal pinhole camera. A perspective projection associated with the focal length f maps a 3-D point M whose coordinates in the camera-centered coordinate system are (X, Y, Z) to an "undistorted" image point $m_u = (x_u, y_u)$ on the image plane:

$$\begin{aligned} x_u &= f\frac{X}{Z} \\ y_u &= f\frac{Y}{Z} \end{aligned} \tag{1}$$

Then, the image distortion transforms m_u to a distorted image point m_d. The image distortion model [22] is usually given as a mapping from the distorted image coordinates, which are observable in the acquired images, to the undistorted image coordinates, which are needed for further calculations. The image distortion function can be decomposed in two terms: radial and tangential distortion. Radial distortion is a deformation of the image along the direction from a point called the center of distortion to the considered image point, and tangential distortion is a deformation perpendicular to this direction. The center of distortion is invariant under both transformations.

It was found that for many machine vision applications, tangential distortion need not to be considered [26]. Let R be the radial distortion function, which is invertible over the image:

$$R : r_u \longrightarrow r_d = R(r_u), \text{with } \frac{\partial R}{\partial r_u}(0) = 1 \tag{2}$$

The distortion model can be written as:

$$x_u = x_d \frac{R^{-1}(r_d)}{r_d},\ y_d = y_d \frac{R^{-1}(r_d)}{r_d} \tag{3}$$

where $r_d = \sqrt{x_d^2 + y_d^2}$, and similarly the inverse distortion model is:

$$x_d = x_u \frac{R(r_u)}{r_u},\ y_d = y_u \frac{R(r_u)}{r_u} \tag{4}$$

where $r_u = \sqrt{x_u^2 + y_u^2}$.

Finally, distorted image plane coordinates are converted to frame buffer coordinates, which can be expressed either in pixels or in normalized coordinates (i.e. pixels divided by image dimensions), depending on the unit of f:

$$\begin{aligned} x_i &= S_x x_d + C_x \\ y_i &= y_d + C_y \end{aligned} \tag{5}$$

where (C_x, C_y) are the image coordinates of the principal point and S_x is the image aspect ratio.

In our case we want to decouple the effect of distortion from the projection on the image plane, because we want to calibrate is the distortion without knowing anything about internal camera parameters. Consequently, in our model, the center of distortion (c_x, c_y) will be different from the principal point (C_x, C_y). It was shown [23] that this is mainly equivalent to adding decentering distortion terms to the distortion model of equation 6. A higher order effect of this is to apply an (very small) affine transformation to the image,

but the affine transform of a pinhole camera is also a pinhole camera (i.e. this is a linear distortion effect).

Moreover, the image aspect ratio s_x that we use in the distortion model may not be the same as the real camera aspect ratio S_x. The difference between these two aspect ratios will result in another term of tangential distortion. To summarize, the difference between the coordinates of the center of distortion (c_x, c_y) and those of the principal point (C_x, C_y) corresponds to decentering distortion because the center of distortion may be different from principal point, and the difference between the distortion aspect ratio s_x and the camera aspect ratio S_x corresponds to a term of tangential distortion.

In the following, all coordinates are frame buffer coordinates, either expressed in pixels or normalized (by dividing x by the image width and y by the image height) to be unit-less.

2.1 Polynomial distortion models

The lens distortion model (equations 3) can be written as an infinite series:

$$\begin{aligned} x_u &= x_d(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + \cdots) \\ y_u &= y_d(1 + \kappa_1 r_d^2 + \kappa_2 r_d^4 + \cdots) \end{aligned} \tag{6}$$

Several tests [3,26] showed that using only the first order radial symmetric distortion parameter κ_1, one could achieve an accuracy of about 0.1 pixels in image space using lenses exhibiting large distortion, together with the other parameters of the perspective camera [13].

The undistorted coordinates are given by the formula:

$$\begin{aligned} x_u &= x_d(1 + \kappa_1 r_d^2) \\ y_u &= y_d(1 + \kappa_1 r_d^2) \end{aligned} \tag{7}$$

where $r_d = \sqrt{x_d^2 + y_d^2}$ is the distorted radius.

The inverse distortion model is obtained by solving the following equation for r_d, given r_u:

$$r_u = r_d \left(1 + \kappa_1 r_d^2\right) \tag{8}$$

where $r_u = \sqrt{x_u^2 + y_u^2}$ is the undistorted radius and r_d is the distorted radius.

This is a polynomial of degree three in r_d of the form $r_d^3 + cr_d + d = 0$, with $c = \frac{1}{\kappa_1}$ and $d = -cr_u$, which can be solved using the Cardan method which is a direct method for solving polynomials of degree three. It has either one or three real solutions, depending on the sign of the discriminant:

$$\Delta = Q^3 + R^2$$

where $Q = \frac{c}{3}$ and $R = -\frac{d}{2}$.

If $\Delta > 0$ there is only one real solution:

$$r_d = \sqrt[3]{R + \sqrt{\Delta}} + \frac{Q}{\sqrt[3]{R + \sqrt{\Delta}}} \tag{9}$$

and if $\Delta < 0$ there are three real solutions but only one is valid because when r_u is fixed, r_d must be a continuous function of κ_1. The continuity at $\kappa_1 = 0$ gives the solution:

$$r_d = -S \cos T + S\sqrt{3} \sin T \tag{10}$$

where $S = \sqrt[3]{\sqrt{R^2 - \Delta}}$ and $T = \frac{1}{3} \arctan \frac{\sqrt{-\Delta}}{R}$

Combining equations 7 and 8, the distorted coordinates are given by:

$$\begin{aligned} x_d &= x_u \frac{r_d}{r_u} \\ y_d &= y_u \frac{r_d}{r_u} \end{aligned} \tag{11}$$

With high-distortion lenses, it may be necessary to include higher order terms of Equation 6 in the distortion model [17]. In this case, the transformation from undistorted to distorted coordinates has no closed-from solution, and a line solver has to be used (a simple Newton method is enough).

In the case of fish-eye lens and some other high-distortion lens, nonlinear distortion was built-in on purpose, in order to correct deficiencies of wide-angle distortion-free lens, such as the fact that objets near the border of the field-of-view have an exagerated size on the image. To model the distortion of these lens, it may be necessary to take into account many terms of Equation 6: in our experiences, distortion models of order at least 3 (which correspond to a seventh order polynomial for radial distortion) had to be used to compensate for nonlinear distortion of fish-eye lens. For this reason, we looked for distortion models which are more suitable to this kind of lens.

2.2 Fish-eye models

Fish-eye lenses are designed from the ground up to include some kind of nonlinear distortion. For this reason, it is better to use a distortion model that tries to mimic this effect, rather than to use a high number of terms in the series of Equation 6. Shah and Aggarwal [21] showed that when calibrating a fish-eye lens using a 7th order odd powered polynomial for radial distortion (which corresponds to a third order distortion model), distortion still remains, so that they have to use a model with even more degrees of freedom.

Basu and Licardie [1] use a logarithmic distortion model (FET, or Fish-Eye Transform) or a polynomial distortion model (PFET) to model fish-eye lenses, and the PFET model seems to perform better than the FET. The FET model is based on the observation that fish-eye have a high resolution at the fovea, and a non-linearly decreasing resolution towards the periphery. The corresponding radial distortion function is:

$$r_d = R(r_u) = s \log\left(1 + \lambda r_u\right) \tag{12}$$

We propose here another distortion model for fish-eye lens, which is based on the way fish-eye lenses are designed: The distance between an image point and the principal point is usually roughly proportional to the angle between the corresponding 3-D point, the optical center and the optical axis (Figure 1), so that the angular resolution is roughly proportional to the image resolution along an image radius. This model has only one parameter, which is the field-of-view ω of the corresponding *ideal* fish-eye lens, so we called it the

FOV model. This angle may not correspond to the real camera field-of-view, since the fish-eye optics may not follow exactly this model. The corresponding distortion function and its inverse are:

$$r_d = \frac{1}{\omega} \arctan\left(2 r_u \tan\frac{\omega}{2}\right), \tag{13}$$

$$\text{and } r_u = \frac{\tan(r_d \omega)}{2 \tan\frac{\omega}{2}} \tag{14}$$

If this one-parameter model is not sufficient to model the complex distortion of fish-eye lens, the previous distortion model (Equation 6) can be applied before Equation 14, with $\kappa_1 = 0$ (ω, as a first order distortion parameter, would be redundant with κ_1). A second order FOV model will have $\kappa_2 \neq 0$, and a third order FOV model will have $\kappa_3 \neq 0$.

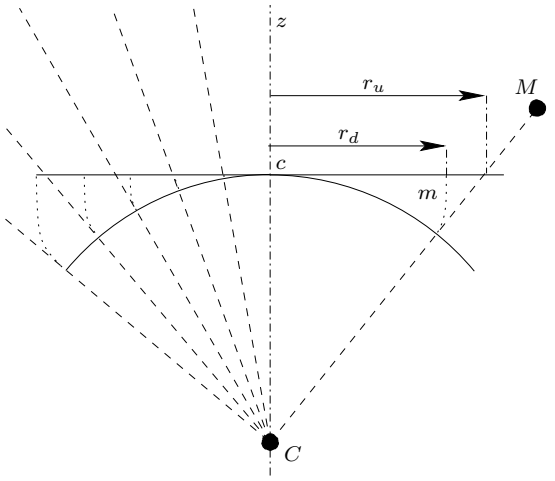

Fig. 1 In the FOV distortion model, the distance cm is proportional to the angle between (CM) and the optical axis (Cz)

2.3 Inverse models

Using the models described before, the cheapest transformation in terms of calculation is from the the distorted coordinates to undistorted coordinates. This also means that it is cheaper to detect features in the distorted image and to undistort them, than to undistort the whole image and to extract the feature from the undistorted image: in fact, undistorting a whole image consists of computing the distorted coordinates of every point in the undistorted image (which requires solving a third degree polynomial –for the first-order model– or more complicated equations), and then computing its intensity value by bilinear interpolation in the original distorted image.

For some algorithms or feature detection methods which depend on linear perspective projection images, one must nevertheless undistort the whole image. A typical example is stereo by correlation, which require an accurate rectification of images. In these cases, where calibration time may not be crucial but images need to be undistorted quickly (i.e. only the transform function from undistorted to distorted coordinated is to be used more often than its inverse in a program's main loop), then a good solution is to switch the distortion function and its inverse. For the first order distortion model, Equation 7 would become the distortion function and equation 11 its inverse. This is what we call an order -1 polynomial model in this paper. That way the automatic distortion calibration step is costly (because, as we will see later, it requires undistorting edge features), but once the camera is calibrated, the un-distortion of the whole intensity images is a lot faster.

Inverse model can be derived from polynomial models, fish-eye models, FOV model, or any distortion model. Though they have the same number of parameters as their direct counterpart, we will see section 5.4 that they do not represent the same kind of distortion, and may not be able to deal with agiven lens distortion.

3 Distortion calibration

3.1 Principle of distortion calibration

The goal of the distortion calibration is to find the transformation (or un-distortion) that maps the actual camera image plane onto an image following the perspective camera model. To find the distortion parameters described in section 2, we use the following fundamental property: a camera follows the perspective camera model if and only if the projection of *every* 3-D line in space onto the camera plane is a line. Consequently, all we need is a way to find projections of 3-D lines in the image (they are not lines anymore in the images, since they are distorted, but curves), and a way to measure how much each 3-D line is distorted in the image. Then we will just have to let the distortion parameters vary, and try to minimize the distortion of edges transformed using these parameters.

3.2 Edge detection with sub-pixel accuracy

The first step of the calibration consists of extracting edges from the images. Since image distortion is sometimes less than a pixel at image boundaries, there was definitely a need for an edge detection method with a sub-pixel accuracy. We developed an edge detection method [11], which is a sub-pixel refinement of the classical Non-Maxima Suppression (NMS) of the gradient norm in the direction of the gradient. It was shown to give edge position with a precision varying from 0.05 pixel RMS for a noise-free synthetic image, to 0.3 pixel RMS for an image Signal to Noise Ratio (SNR) of 18dB (which is actually a lot of noise, the VHS videotapes SNR is about 50dB). In practice, any edge detection method with sub-pixel accuracy can be used.

3.3 Finding 3-D segments in a distorted image

In order to calibrate distortion, we must find edges in the image which are most probably images of 3-D segments. The goal is not to get all segments, but to find the most probable ones. For this reason, we do not care if a long segment, because of its distortion, is broken into smaller segments.

Therefore, and because we are using a subpixel edge detection method, we use a very small tolerance for polygonal approximation: the maximum distance between edge points and the segment joining both ends of the edge must typically be less than 0.4 pixels. We also put a threshold on segment length of about 60 pixels for a 640×480 image, because small segments may contain more noise than useful information about distortion.

Moreover, because of the corner rounding effect [8,9] due to edge detection, we throw out a few edgels (between 3 and 5, depending on the amount of smoothing performed on the image before edge detection) at both ends of each detected edge segment.

3.4 Measuring distortion of a 3-D segment in the image

In order to find the distortion parameters we use a measure of how much each detected segment is distorted. This distortion measure will then be minimized to find the best calibration parameters. One could use for example the mean curvature of the edges, or any distance function on the edge space that would be zero if the edge is a perfect segment and the more the segment would be distorted, the bigger the distance would be.

We chose a simple measure of distortion which consists of doing a least squares approximation of each edge which should be a projection of a 3-D segment by a line [10], and to take for the distortion error the sum of squares of the distances from the point to the line (i.e. the χ^2 of the least square approximation, Figure 2). That way, the error is zero if the edge lies exactly on a line, and the bigger the curvature of the edge, the bigger the distortion error.

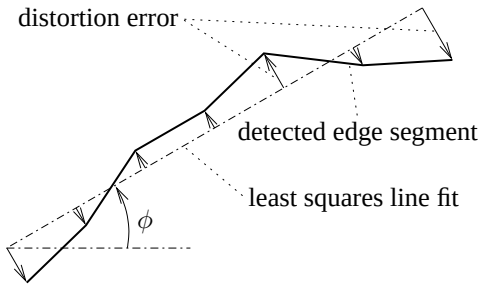

Fig. 2 The distortion error is the sum of squares of the distances from the edgels of an edge segment to the least square fit of a line to these edgels .

This leads to the following expression for the distortion error of each edge segment [10]:

$$\chi^2 = a \sin^2 \phi - 2\,|b|\,|\sin\phi| \cos\phi + c \cos^2 \phi \tag{15}$$

where:

$$a = \sum_{j=1}^{n} x_j^2 - \frac{1}{n}(\sum_{j=1}^{n} x_j)^2 \tag{16}$$

$$b = \sum_{j=1}^{n} x_j y_j - \frac{1}{n} \sum_{j=1}^{n} x_j \sum_{j=1}^{n} y_j \tag{17}$$

$$c = \sum_{j=1}^{n} y_j^2 - \frac{1}{n}(\sum_{j=1}^{n} y_j)^2 \tag{18}$$

$$\alpha = a - c;\ \beta = \frac{\alpha}{2\sqrt{\alpha^2 + 4b^2}} \tag{19}$$

$$|\sin\phi| = \sqrt{1/2 - \beta};\ \cos\phi = \sqrt{1/2 + \beta} \tag{20}$$

ϕ is the angle of the line in the image, and $\sin\phi$ should have the same sign as b. ϕ can also be computed as $\phi = 1/2 \arctan 2(2b, a - c)$, but only $\sin\phi$ and $\cos\phi$ are useful to compute χ^2.

3.5 Putting it all together: The whole calibration process

The whole distortion calibration process is not done in a single step (edge detection, polygonal approximation, and optimization), because there may be outliers in the segments detected by the polygonal approximation, i.e. segment edges which do not really correspond to 3-D line segments. Moreover, some images of 3-D line segments may be broken into smaller edges because the first polygonal approximation is done on distorted edges. By doing another polygonal approximation after the optimization, on undistorted edges, we can eliminate many outliers easily and sometimes get longer segments which contain more information about distortion. This way we get even more accurate calibration parameters.

A first version of the distortion calibration process is:

1. Load or acquire a set of images.
2. Do subpixel edge detection and linking on all the images in the collection. The result is the set of linked edges of all images.
3. Initialize the distortion parameters with reasonable values.
4. Do polygonal approximation on undistorted edges to extract segment candidates.
5. Compute the distortion error $E_0 = \sum \chi^2$ (sum is done over all the detected segments).
6. Optimize the distortion parameters κ_1, c_x, c_y, s_x to minimize the total distortion error. The total distortion error is taken as the sum of the distortion errors (eq. 15) of all detected line segments, and is optimized using a nonlinear least-squares minimization method (e.g. Levenberg-Marquart).
7. Compute the distortion error E_1 for the optimized parameters.

8. If the relative change of error $\frac{E_0 - E_1}{E_1}$ is less than a threshold, stop here.
9. update the distortion parameters with the optimized values.
10. Go to step 4.

By minimizing on all the parameters when the data still contains many outliers, there is a risk of getting farther from the optimal parameters. For this reason, steps 3 to 9 are first done with optimization only on the first radial distortion parameter (κ_1 for polynomial models, ω for FOV models) until the termination condition of step 8 is verified, then c_x and c_y are added, and finally full optimization on the distortion parameters (including s_x) is performed. During the process, polygonal approximation (step 4) progressively eliminates most outliers.

Of course, the success of the whole process depends on the number, length and accuracy of the line segments detected in the images. Moreover, the segments should be have various positions and orientations in the image, in order to avoid singular or almost singular situations. For example, one cannot compute radial distortion if all straight lines supporting the detected segments go through a single point in the image. Fortunately, data is cheap in our case, since getting more line segments usually involves only moving the camera and taking more pictures. Instead of analyzing how the number, length and accuracy of the detected segments influence the stability and accuracy of the algorithm, we judged that there was enough data if adding more data (i.e. more pictures) wouldn't change the results significantly. A more in-depth study on what minimum data is necessary for the calibration would be useful, especially in situations where "getting more data" is a problem.

4 Model selection

We have shown that several models can be used to describe a lens' nonlinear distortion: polynomial distortion models (eq. 6) with differents orders (first order uses only κ_1, second order κ_1 and κ_2, etc.), fish-eye models such as the FET model (eq. 12) or the FOV model (eq. 14) with different orders (first order uses only ω, second order is the application of a first order polynomial model before eq. 14, etc.), but then arises the problem of chosing the right model for a given lens.

4.1 Probabilistic approach

The easiest way of chosing the model that best describes some data, based on probability theory, is to take the one that gives the lowest residuals. This usually leads to picking the model with the biggest number of parameters, since increasing the number of parameters usually lowers the residuals (an extreme case is when there is as many parameters as residuals, and the residuals can be zero). In the experimental setup we used, the models have a reduced number of parameters (at most 6 for order 3 models), and we can get as much data as we want (data is edges in our case), simply by acquiring more images with the same lens (the scene need not to be different for each image, moving around the camera is enough). For a given kind of model (e.g. polynomial), this method will almost always pick the model with the highest number of parameters, but we will still be able to say, between two different kinds of models with a given number of parameters, for example a third order polynomial model and a third order FOV model, which one is best. We will also be able to state how much more accuracy we get by adding one order to a given model.

4.2 MDL et al.

When the number of images is limited, or the camera is fixed (e.g. a surveillance camera), a smarter selection method should be used. A proper model selection method would be based on the fact that the model that best describes the data leads to the shortest encoding (or description, in the information theory sense) of the model and the data. This principle is called *Minimum Description Length* [20,14], and is now widely used in computer vision. The MDL principle, or other model selection methods based on information theory [25,16] require a fine analysis of the properties of the data and the model, which are beyond the scope of this paper.

When the amount of data (edges in our case) increases, these approaches become asymptotically equivalent to the probabilistic method, because almost all information is contained in the data. A different way of understanding this is that in the ideal case where we have an infinite number of data with unbiased noise, the best model will always be the one that gives the lowest residuals, whereas with only a few data, the model that gives the lowest residuals may fit the noise instead of the data itself. For this reason, we used the simpler probabilistic method in our experiments, because we can gen as much data as we want, just by using edges extracted from additional images taken with the same lens.

4.3 Conversion between distortion models

Suppose we have selected the distortion model that best fits our lens, then one may want to know how much accuracy is lost, in terms of pixel displacement, when using another model instead of this one. Similarly, if two models seem to perform equally with respect to our distortion calibration method, one may want to be able to measure how much *geometrically* different these models are. To answer these questions, we developped a conversion method that picks within a distortion model family the one that most resembles a given model from another family, and also measures how much different these models are.

One way to measure how close distortion model A with parameters p_A is to distortion model B is to try to convert the parameter set describing the first model to a parameter

set p_B describing the second model. Because the two models belong to different families of transformations, this conversion is generally not possible, but we propose the following method to get the parameter set p_B of model B which gives the best approximation of model $A(p_A)$ the least square sense.

The parameter set p_B is chosen so that the distorted image is undistorted "the same way" by $A(p_A)$ and $B(p_B)$. "The same way" means that there is at most a non-distorting transformation between the set of points undistorted by $A(p_A)$ and the set of points undistorted by $B(p_B)$. We define a non-distorting transformation as being linear in projective coordinates. The most general non-distorting transformation is an homography, so we are looking for parameters p_B and a homography H so that the image undistorted by $B(p_B)$ and transformed by H is as close as possible to the image undistorted by $A(p_A)$.

Let us consider an infinite set of points $\{m_i\}$ uniformly distributed on the distorted image, let $\{m_i^A\}$ be these points undistorted using $A(p_A)$, and let $\{m_i^B\}$ be the same points undistorted by $B(p_B)$. We measure the closeness[1] from $A(p_A)$ to $B(p_B)$ as:

$$C(A(p_A), B(p_B)) = \sqrt{\inf_H \lim_{i\to\infty} \frac{1}{i} \sum_i \left| m_i^A - H(m_i^B) \right|^2} \tag{21}$$

The conversion from $A(p_A)$ to model B is simply achieved by computing p_B (and H) that minimize (21), i.e. $p_B = \arg\inf_{p_B} C(A(p_A), B(p_B))$. In practice, of course, we use a finite number of uniformly distributed points (e.g. 100×100).

$C(A(p_A), B(p_B))$ is the mean residual error in image coordinates units, and measures how good model B fits to $A(p_A)$. This can be used, if B has less parameters than A, to check if $B(p_B)$ is good enough to represent the distortion yielded by $A(p_A)$. An example is shown on fish-eye lens in section 5.4.

5 Results and comparison with a full calibration method

5.1 Experimental setup

We used various hardware setups to test the accuracy of the distortion calibration, from low-cost video-conference video hardware to high-quality cameras and frame-grabber.

The lowest quality hardware is a very simple video acquisition system included with every Silicon Graphics Indy workstation. This system is not designed for accuracy nor quality and consists of an IndyCam camera coupled with the standard Vino frame grabber. The acquired image is 640×480 pixels interlaced, and contains a lot of distortion and blur caused by the cheap wide-angle lens. The use of an on-line camera allows very fast image transfer between the frame grabber and the program memory using Direct Memory Access (DMA), so that we are able to do fast distortion calibration. The quality of the whole system seems comparable to that of a VHS videotape.

Other images were acquired using an Imaging Technologies acquisition board together with several different camera setups: a Sony XC75CE camera with 8mm, 12.5mm, and 16mm lens (the smaller the focal length, the more important the distortion), and an old Pulnix TM-46 camera with 8mm lens. The fish-eye images come from a custom underwater camera[2]

The distortion calibration software is a stand-alone program that can either work on images acquired on-line using a camera and a frame grabber or acquired off-line and saved to disk. Image gradient was computed using a recursive Gaussian filter [7], and subsequent edge detection was done by NMS.

The optimization step was performed using the subroutine `lmdif` from MINPACK or the subroutine `dnls1` from SLATEC, both packages being available from Netlib[3].

5.2 The full calibration method

In order to evaluate the validity of the distortion parameters obtained by our method, we compared them to those obtained by a method for full calibration (both external and internal) that incorporates comparable distortion parameters. The software we used to do full calibration implements the Tsai calibration method [26] and is freely available. This software implements calibration of external (rotation and translation) and internal camera parameters at the same time. The internal parameter set is composed of the pinhole camera parameters except for the shear parameter (which is very close to zero on CCD cameras anyway [3]), and of the first radial distortion parameter. From the result of this calibration mechanism, we can extract the position of the principal point, the image aspect ratio, and the first radial distortion parameter.

As seen in section 2, though, these are not exactly the same parameters as those that we can compute using our method, since we allow more degrees of freedom for the distortion function: two more parameters of decentering distortion and one parameter of tangential distortion. Having different coordinates for the principal point and the center of distortion, and for the image aspect ratio and distortion aspect ratio. There are two ways of comparing the results of the two methods: one is to compute the closeness (defined in sec. 4.3) between the two sets of parameters by computing the best homography between two sets of undistorted points, the other is to convert the radial distortion parameter found by Tsai cali-

[1] This measure is not a distance, since $C(A(p_A), B(p_B)) \neq C(B(p_B), A(p_A))$. A distance derived from C is $C'(A(p_A), B(p_B)) = \sqrt{C^2(A(p_A), B(p_B)) + C(B(p_B), A(p_A))}$, but our measure reflects the fact that "finding the best parameters p_B for model B to fit model $A(p_A)$" is a non-symmetric process.

[2] Thanks go to J. Ménière and C. Migliorini from Poseidon, Paris, who use these cameras for swimming-pool monitoring, for letting me use these images.

[3] http://www.netlib.org/

bration using the distortion center and aspect ratio found by our method, and vice-versa.

5.3 Results

We calibrated a set of cameras using the Tsai method and a calibration grid (Figure 3) with 128 points, and we computed the distortion parameters from the result of this full calibration method (Table 2) (camera E could not be calibrated this way, because the automatic feature extraction used before Tsai calibration didn't work on these images). The distortion calibration method was also applied to sets of about 30 images (see Figure 4) for each camera/lens combination, and the results for the four parameters of distortion are shown in Table 1. For each set of images, the edges extracted from all the images are used for calibration. The initial values for the distortion parameters before the optimization were set to "reasonable" values, i.e. the center of distortion was set to the center of the image, κ_1 was set to zero, and s_x to the image aspect ratio, computed for the camera specifications. For the IndyCam, this gave $c_x = c_y = \frac{1}{2}$, $\kappa_1 = 0$ and $s_x = \frac{3}{4}$.

All the parameters and results are given in normalized coordinates and are dimensionless: x is divided by the image width and y by the image height, thus $(x, y) \in [0, 1]^2$. This way, we are able to measure and compare the effect of the lens, and we are as independant as possible of the frame grabber (the results presented here were obtained using various frame grabbers).

As explained in section 2, these have not exactly the same meaning as the distortion parameters obtained from Tsai calibration, mainly because in this model the distortion center is the same as the optical center, and also because we introduced a few more degrees of freedom in the distortion function, allowing decentering and tangential distortion. This explains why the distortion center found on low-distortion cameras such as the Sony 16mm are so far away from the principal point.

The four bottom lines of Table 1 may need some more explanations. C_0 is the closeness between the computed distortion model and a zero-distortion camera model (i.e. $\kappa_1 = 0$). It is a good way to measure *how much* distorting this model is: for example, for camera A, C_0 is $6.385 \cdot 10^{-3}$ in normalized coordinates, which corresponds to about 4 pixels of "mean distortion" over the image (not only in the corners!). The measure C_t says that there is about 0.5 pixels RMS between the distortion model computed with our method and the Tsai model, for all camera/lens combinations, which means that the quality of our method is intrinsically acceptable, but there is still no way to tell which of both method, Tsai or automatic, gives best results. For cameras with high distortion, like the IndyCam and the cameras with 8mm lens, The center of distortion and the distortion aspect ratio are close to the principal point and the image aspect ratio computed by Tsai calibration.

The seventh line of Table 1 gives the result of the conversion from the Tsai set of parameters $(c'_x, c'_y, s'_x, \kappa'_1)$ (Table 2) to a model where the center and aspect ratio of distortion are fixed to the values c_x, c_y, s_x. The resulting set of parameters is $(c_x, c_y, s_x, \psi(\kappa'_1))$, and the RMS residual error of the convertion, i.e. the closeness

$$C_f = C((c_x, c_y, s_x, f(\kappa'_1)), (c'_x, c'_y, s'_x, \kappa'_1)),$$

is always below one third of a pixel (last line of the table), which allows us to compare $f(\kappa'_1)$ with κ_1. In fact, for all camera configurations, parameter $f(\kappa'_1)$ is close to κ_1 obtained by our automatic calibration method, meaning that, once again, our results are very close to those given by Tsai calibration, though they *look* different (especially for low-distortion lenses).

Figure 5 shows a sample image, before and after the correction. This image was affected by pin-cushion distortion, corresponding to a positive value of κ_1. Barrel distortion corresponds to negative values of κ_1.

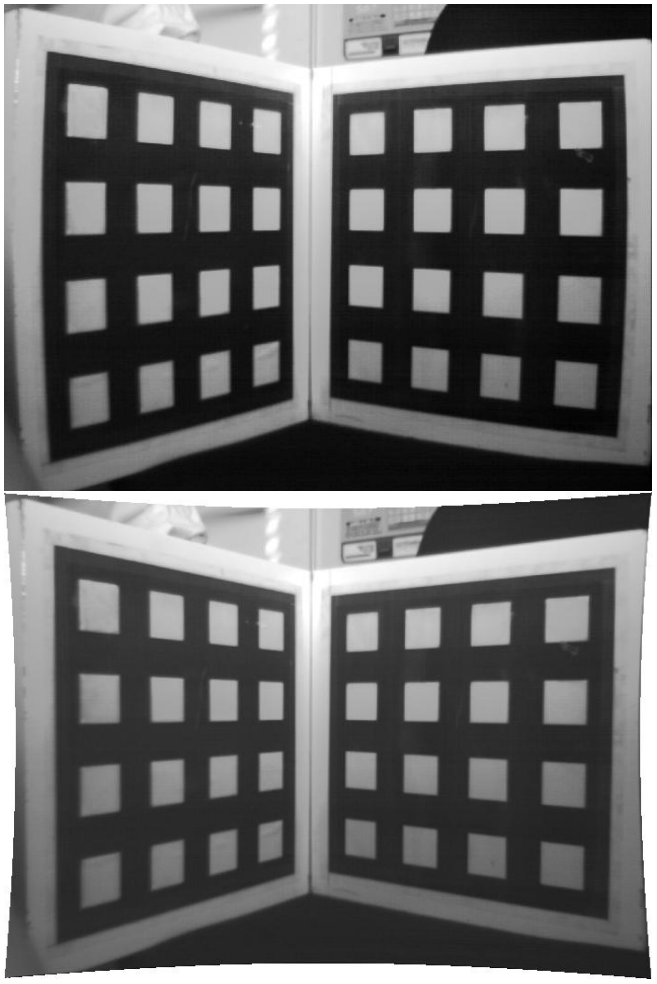

Fig. 3 The calibration grid used for Tsai calibration: original distorted image (top) and image undistorted using the parameters computed by our method (bottom).

camera/lens	A	B	C	D	E
c_x	0.493	0.635	0.518	0.408	0.496
c_y	0.503	0.405	0.122	0.205	0.490
s_x	0.738	0.619	0.689	0.663	0.590
κ_1	0.154	0.041	0.016	0.012	-0.041
$C_0.10^3$	6.385	2.449	1.044	0.770	2.651
$C_t.10^3$	0.751	0.923	0.811	0.626	N/A
$\psi(\kappa_1')$	0.137	0.028	0.004	0.002	N/A
$C_f.10^3$	0.217	0.516	0.263	0.107	N/A

Table 1 The distortion parameters obtained on various camera/lens setups using our method, in normalized image coordinates: First radial distortion parameter κ_1, position of the center of distortion c_x, c_y, and distortion aspect ratio s_x (not necessarily the same as S_x, the image aspect ratio). C_0 is the closeness with a zero-distortion model, C_t is the closeness between these parameters and the ones found by Tsai calibration, $\psi(\kappa_1')$ is the first order radial distortion converted from results of Tsai calibration (Table 2) using the distortion center (c_x, c_y) and aspect ration s_x from our method. C_f is the RMS residual error of the convertion. All parameters are dimensionless. A is IndyCam, B is Sony XC-75E camera with 8mm lens, C is Sony XC-75E camera with 12.5mm lens, D is Sony XC-75E with 16mm lens, E is Pulnix camera with 8mm lens.

camera/lens	A	B	C	D
c_x'	0.475	0.514	0.498	0.484
c_y'	0.503	0.476	0.501	0.487
s_x'	0.732	0.678	0.679	0.678
κ_1'	0.135	0.0358	0.00772	0.00375

Table 2 The distortion parameters obtained using the Tsai calibration method, in normalized image coordinates: position of the principal point, and image aspect ratio, First radial distortion parameter. See Table 1 for details on the camera/lens configurations.

Fig. 4 Some of the images that were used for distortion calibration. Even blurred or fuzzy pictures can be used.

Fig. 5 A distorted image with the detected segments (left) and the same image at the end of the distortion calibration with segments extracted from undistorted edges (right): some outliers (wrong segments on the plant) were removed and longer segments are detected. This image represents the worst case, where some curves may be mistaken for lines.

5.4 Choice of the distortion model

In this experiment, we use an underwater fish-eye lens (Figure 6), and we want to find which distortion model best fits this lens. Besides, we will try to evaluate which order of radial distortion is necessary to get a given accuracy. The distortion models tested on this lens are FOV1, FOV2, FOV3 (first, second, and third order FOV models), P1, P2, P3 (first, second and third order polynomial models), P-1, P-2, P-3 (first, second and third order inverse polynomial models).

Results (Table 3) show for each model the total number of segments detected at the end of the calibration stage, the number of edgels forming these segments, and the mean edgel distortion error (Equation 15) in normalized image coordinates.

The image size is 384×288, and we notice that the mean edgel distortion error is almost the same for all models, and comparable to the theoretical accuracy of the edge detection method (0.1 pixel) [11]. We can judge the quality of the distortion models from the number of detected edgels which

Fig. 6 An image taken with underwater fish-eye lens, and the same image undistorted using model 3^f. The parallel lines help checking the result of distortion calibration.

model	nb. seg.	nb. edgels	seg. len.	dist. err.$\cdot 10^3$
FOV1	591	78646	133.1	0.312
FOV2	589	77685	131.9	0.298
FOV3	585	79718	136.3	0.308
P1	670	71113	106.1	0.304
P2	585	77161	131.9	0.318
P3	588	77154	131.2	0.318
P-1	410	48352	117.9	0.300
P-2	534	68286	127.9	0.308
P-3	549	71249	129.8	0.312

Table 3 The results of calibration on the same set of images using different distortion models. Number of segments detected, number of edgels forming these segments, mean segment length, and mean edgel distortion error.

were classified as segments, and from the mean segment length. From these, we can see that model FOV3 gives the best results, and that all versions of the FOV model (FOV1, FOV2, FOV3) perform better than polynomial models (P1, P2, P3) for this lens. We also notice that inverse polynomial models (P-1, P-2, P-3) perform poorly, compared with their direct counterpart. From these measurements, we clearly see that though they have the same number of degrees of freedom, different distortion models (eg. FOV3, P3 and P-3) describe more or less accurately the real distortion transformation. Therefore, the distortion model must be chosen carefully.

Once we have chosen the distortion model (FOV in this case), we still have to determine what order is necessary to get a given accuracy. For this, we use the residual error of the conversion from the highest-order model to a lower order model (section 4.3). These residuals, conputed for conversions from FOV3 and P3 models, are shown Table 4.

from \ to	FOV1	FOV2	FOV3	P1	P2	P3
FOV3	0.69	0.10	N/A	2.00	0.21	0.03
P3	1.00	0.04	0.02	2.08	0.03	N/A

from \ to	P-1	P-2	P-3
FOV3	7.29	1.32	1.16

Table 4 Residual errors in 10^{-3} normalized coordinates after converting from models FOV3 and P3 to other models.

From these results, we immediately notice that inverse polynomial models (P-1, P-2, P-3) are completely inadequate for this lens, since they can lead to mean distortion errors from one half to several pixels, depending on the order, but we already noticed that these models were not suitable from the calibration results (Table 3).

The most important result is that by using FOV2 instead of FOV3, we will get a mean distortion error of about 0.2 pixels (for a 512×512 image), if we use P2 this error will be 0.4 pixels, and if we use FOV1 it will be 1.4 pixels. Consequently, if we need the best accuracy, we have to use the FOV3 model, but FOV2 and P2 represent a good compromise between performance and accuracy. FOV2 is especially interesting, since a closed form inverse function is available for this model.

This investigation was made on our underwater camera, but the same investigation could be made on other lenses. This could, of course, lead to a *different optimal distortion model* than FOV2, but the method would be the same.

6 Discussion

With computer vision applications demanding more and more accuracy in the camera model and the calibration of its parameters, there is definitely a need for calibration methods that don't rely on the simple projective linear pinhole camera model. Camera optics still have lots of distortion, and zero-distortion wide-angle lens exist but remain very expensive.

The automatic distortion calibration method presented here has many advantages over other existing calibration methods that use a camera model with distortion [3,4,24,26]. First, it makes very few assumptions on the observed world: there is no need for a calibration grid [3,4,26]. All it needs is images of scenes containing 3-D segments, like interior scenes or city scenes. Second, it is completely automatic, and camera motion needs not to be known [23,24]. It can even be applied to images acquired off-line, which could come from a surveillance videotape or a portable camcorder. Results of distortion calibration and comparison with a grid-based calibration method [18] are shown for several lenses and cameras.

If we decide to calibrate distortion, there is not a unique solution for the choice of the kind of distortion model [1] and the order of this distortion model. For example, fish-eye lens may not be well represented by the traditional polynomial distortion model. We presented an alternative fish-eye model, called the FOV model, together with methods to determine which model is best for a given lens, and at which order. This study was made in the case of an underwater fish-eye camera, and the results showed that the highest-order model may not always be necessary, depending on the required accuracy, and that different models with the same number of parameters don't necessarily give the same accuracy.

Once the distortion is calibrated, any computer vision algorithm that relies on the pinhole camera model can be used, simply by applying the inverse of the distortion either to image features (edges, corners, etc.) or to the whole image. This method could also be used together with self-calibration or weak calibration methods that would take into account the distortion parameters. The distortion calibration could be done before self-calibration, so that the latter would use un-distorted features and images, or during self-calibration[28], the distortion error being taken into account in the self-calibration process.

References

1. Anup Basu and Sergio Licardie. Alternative models for fish-eye lenses. *Pattern Recognition Letters*, 16:433–441, April 1995.
2. P. Beardsley, D. Murray, and A. Zissermann. Camera calibration using multiple images. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision*, pages 312–320, Santa Margherita, Italy, May 1992. Springer-Verlag.
3. Horst A. Beyer. Accurate calibration of CCD-cameras. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, Urbana Champaign, IL, June 1992. IEEE.
4. P. Brand, R. Mohr, and P. Bobet. Distorsions optiques : correction dans un modèle projectif. Technical Report 1933, LIFIA–INRIA Rhône-Alpes, 1993.
5. Duane C. Brown. Close-range camera calibration. *Photogrammetric Engineering*, 37(8):855–866, 1971.
6. Bruno Caprile and Vincent Torre. Using Vanishing Points for Camera Calibration. *The International Journal of Computer Vision*, 4(2):127–140, March 1990.
7. R. Deriche. Recursively implementing the gaussian and its derivatives. Technical Report 1893, INRIA, Unité de Recherche Sophia-Antipolis, 1993.
8. R. Deriche and G. Giraudon. Accurate corner detection: An analytical study. In *Proceedings of the 3rd International Conference on Computer Vision*, pages 66–70, Osaka, Japan, December 1990. IEEE Computer Society Press.
9. R. Deriche and G. Giraudon. A computational approach for corner and vertex detection. *The International Journal of Computer Vision*, 10(2):101–124, 1993.
10. Rachid Deriche, Régis Vaillant, and Olivier Faugeras. From noisy edge points to 3D reconstruction of a scene: A robust approach and its uncertainty analysis. In *Proceedings of the 7th Scandinavian Conference on Image Analysis*, pages 225–232, Alborg, Denmark, August 1991.
11. Frédéric Devernay. A non-maxima suppression method for edge detection with sub-pixel accuracy. RR 2724, INRIA, November 1995.
12. Olivier Faugeras, Tuan Luong, and Steven Maybank. Camera self-calibration: theory and experiments. In G. Sandini, editor, *Proceedings of the 2nd European Conference on Computer Vision*, pages 321–334, Santa Margherita, Italy, May 1992. Springer-Verlag.
13. Olivier Faugeras and Giorgio Toscani. Structure from Motion using the Reconstruction and Reprojection Technique. In *IEEE Workshop on Computer Vision*, pages 345–348, Miami Beach, November-December 1987. IEEE Computer Society.
14. P. Fua and A.J. Hanson. An optimization framework for feature extraction. *Machine Vision and Applications*, (4):59–87, 1991.
15. Richard Hartley. Projective reconstruction and invariants from multiple images. *PAMI*, 16(10):1036–1040, 1994.
16. K. Kanatani. Automatic singularity test for motion analysis by an information criterion. In Bernard Buxton, editor, *Proceedings of the 4th European Conference on Computer Vision*, pages 697–708, Cambridge, UK, April 1996.
17. J.M. Lavest, M. Viala, and M. Dhome. Do we really need an accurate calibration pattern to achieve a reliable camera calibration. In Hans Burkhardt and Bernd Neumann, editors, *Proceedings of the 5th European Conference on Computer Vision*, volume 1 of *Lecture Notes in Computer Science*, pages 158–174, Freiburg, Germany, June 1998. Springer-Verlag.
18. R. K. Lenz and R. Y. Tsai. Techniques for calibration of the scale factor and image center for high accuracy 3-D machine vision metrology. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:713–720, 1988.
19. M. A. Penna. Camera calibration: A quick and easy way to determine the scale factor. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:1240–1245, 1991.
20. J. Rissanen. Minimum description length principle. *Encyclopedia of Statistic Sciences*, 5:523–527, 1987.
21. Shishir Shah and J.K. Aggarwal. Intrinsic parameter calibration procedure for a (high-distortion) fish-eye lens camera with distortion model and accuracy estimation. *Pattern Recog.*, 29(11):1175–1788, 1996.
22. C. C. Slama, editor. *Manual of Photogrammetry*. American Society of Photogrammetry, fourth edition, 1980.
23. Gideon P. Stein. Internal camera calibration using rotation and geometric shapes. Master's thesis, Massachusetts Institute of Technology, June 1993. AITR-1426.
24. Gideon P. Stein. Accurate internal camera calibration using rotation with analysis of sources of error. In *Proceedings of the 5th International Conference on Computer Vision*, Boston, MA, June 1995. IEEE Computer Society Press.
25. P.H.S. Torr and D.W. Murray. Statistical detection of independent movement from a moving camera. *Image and Vision Computing*, 11(4):180–187, 1993.

26. Roger Y. Tsai. A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses. *IEEE Journal of Robotics and Automation*, 3(4):323–344, August 1987.
27. J. Weng, P. Cohen, and M. Herniou. Camera calibration with distortion models and accuracy evaluation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(10):965–980, October 1992.
28. Z. Zhang. On the epipolar geometry between two images with lens distortion. In *International Conference on Pattern Recognition*, volume I, pages 407–411, Vienna, Austria, August 1996.
29. Z. Zhang, R. Deriche, O. Faugeras, and Q.-T. Luong. A robust technique for matching two uncalibrated images through the recovery of the unknown epipolar geometry. *Artificial Intelligence Journal*, 78:87–119, October 1995.